

(19) 日本国特許庁 (J P)

(12) 公表特許公報 (A)

(11) 特許出願公表番号
特表平10-512070

(43) 公表日 平成10年(1998)11月17日

(51) Int.Cl.⁶
G 0 6 F 9/30

識別記号
3 5 0

F I
G 0 6 F 9/30

3 5 0 E

審査請求 未請求 予備審査請求 有 (全 56 頁)

(21) 出願番号	特願平8-519125	(71) 出願人	インテル・コーポレーション アメリカ合衆国 95052 カリフォルニア 州・サンタ クララ・ミッション カレッ ジ プーレバード・2200
(86) (22) 出願日	平成7年(1995)12月1日	(72) 発明者	ベレグ, アレキザンダー イスラエル国・ハイファ・カメリア・ハン ナ ストリート, 38
(85) 翻訳文提出日	平成9年(1997)6月2日	(72) 発明者	ヤーリ, ヤーコブ イスラエル国・ハイファ・ハナディン・ソ ーロ ハナディン・17/2
(86) 国際出願番号	PCT/US95/15713	(74) 代理人	弁理士 山川 政樹 (外5名)
(87) 国際公開番号	WO96/17291		
(87) 国際公開日	平成8年(1996)6月6日		
(31) 優先権主張番号	08/349, 047		
(32) 優先日	1994年12月2日		
(33) 優先権主張国	米国 (US)		

最終頁に続く

(54) 【発明の名称】 複合オペランドのバック演算機能を有するマイクロプロセッサ

(57) 【要約】
プロセッサが、第1のバックドデータを格納する第1のレジスタ(209)と、デコーダ(202)と、機能ユニット(203)とを備える。デコーダは、第1の制御信号と第2の制御信号を受け取る制御信号入力(207)を有する。第1の制御信号はバック演算を示し、第2の制御信号はアンパック演算を示す。機能ユニットはデコーダ(202)とレジスタ(209)に結合されている。機能ユニットは、第1のバックドデータを使用してバック演算とアンパック演算を行うほか、移動演算も行う。

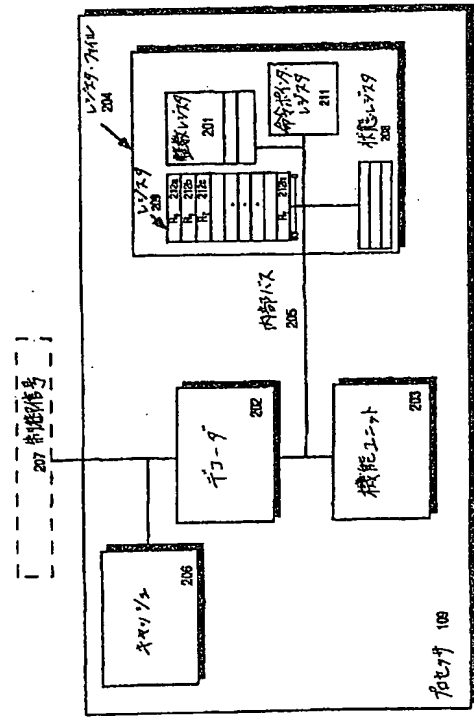


Figure 2

【特許請求の範囲】

1. 第1のパックドデータを格納する第1のレジスタと、
パック演算を示す第1の制御信号とアンパック演算を示す第2の制御信号とを
受け取る制御信号入力と、
前記デコーダと前記第1のレジスタとに結合され、前記第1のパックドデータ
を使用して前記パック演算と前記アンパック演算を実行する機能ユニットとを備
えるプロセッサ。
2. 前記第1のパックドデータが複数のデータ要素を含み、前記複数のデー
タ要素の各データ要素がサイズを有し、前記第1の制御信号が前記サイズに対応す
るインジケータをさらに含むことを特徴とする、請求項1に記載のプロセッサ。
3. 前記サイズがパックドバイトとパックドワードとパックドダブルワードと
のうちの1つであることを特徴とする、請求項2に記載のプロセッサ。
4. 前記第1のパックドデータが64ビットであることを特徴とする、請求項
2に記載のプロセッサ。
5. 前記第1の制御信号がソース・アドレスを含むことを特徴とする、請求項
1に記載のプロセッサ。
6. 前記第1の制御信号が符号インジケータを含み、前記符号インジケータが
前記パック演算を符号付きと無符号のどちらで行うかを決定することを特徴とす
る、請求項1に記載のプロセッサ。
7. 前記制御信号入力がさらに、移動演算を示す第3の制御信号を受け取るこ
とができ、前記プロセッサが第2のレジスタをさらに備え、前記第2のレジスタ
に整数データが格納され、前記移動演算によって前記プロセッサが前記第1のレ
ジスタと前記第2のレジスタの間でデータを移動することを特徴とする、請求項
1に記載のプロセッサ。
8. 前記プロセッサが、レジスタを含むレジスタ・ファイルをさらに備え、前
記第1の制御信号と前記第2の制御信号が第2の場所を含み、前記第2の場所が
前記レジスタに対応し、前記機能ユニットがさらに演算の結果を生成し、前記結
果が前記第2の場所に格納されることを特徴とする、請求項1に記載のプロセッ

サ。

9. 前記プロセッサの第1の制御信号が第2の場所を含み、前記第2の場所が記憶場所に対応し、前記機能ユニットがさらに演算の実行の結果を生成し、前記結果が前記第2の場所に格納されることを特徴とする、請求項1に記載のプロセッサ。

10. デコーダと機能ユニットと第1のレジスタと第2のレジスタとを備え、前記デコーダが前記機能ユニットと前記第1のレジスタと前記第2のレジスタとに結合されたプロセッサにおいてパックドデータを操作する方法であって、

前記デコーダが制御信号をデコードするステップと、

前記制御信号がパック演算を示していると前記デコーダが判断した場合、前記プロセッサが、

前記第1のレジスタに格納されている第1のパックドデータにアクセスするステップと、

前記第2のレジスタに格納されている第2のパックドデータにアクセスするステップと、

前記第1のパックドデータを前記第2のパックドデータと共にパックして結果パックドデータを生成するステップと、

前記結果パックドデータを第1のレジスタに格納するステップとを実行するステップと、

前記制御信号がアンパック演算を示していると前記デコーダが判断した場合、前記プロセッサが、

前記第1のレジスタに格納されている前記第1のパックドデータにアクセスするステップと、

前記第2のレジスタに格納されている前記第2のパックドデータにアクセスするステップと、

前記第1のパックドデータと前記第2のパックドデータをアンパックして前記結果パックドデータを生成するステップと、

前記結果パックドデータを前記第1のレジスタに格納するステップ

とを実行するステップと、

前記制御信号が移動演算を示していると前記デコーダが判断した場合、前記プロセッサが、

前記第2のレジスタに格納されている前記第2のバックドデータにアクセスするステップと、

整数データを記憶する第3のレジスタに前記第2のバックドデータの一部を格納するステップと

を含む、プロセッサにおいてバックドデータを操作する方法。

11. 前記パックが飽和なしで行われ、前記パックが前記第1のバックドデータ内および前記第2のバックドデータ内の各データ要素の下位ビットにアクセスするステップを含み、前記下位ビットが前記結果バックドデータに含まれることを特徴とする、請求項10に記載の方法。

12. 前記第1のバックドデータが第1の複数のデータ要素を含み、前記第1の複数のデータ要素の各データ要素が第1の所定のビット数で表され、前記制御信号がサイズインジケータを含み、前記サイズインジケータが前記第1の所定のビット数を示し、前記結果バックドデータが第2の複数のデータ要素を含み、前記第2の複数のデータ要素が前記第1の複数のデータ要素の2倍のデータ要素を有し、前記第2の複数のデータ要素の各データ要素が第2の所定のビット数で表され、前記第2の所定のビット数が前記第1のビット数の半分であることを特徴とする、請求項10に記載の方法。

13. 前記第1のバックドデータが第1の組のデータ要素を含み、前記第2のバックドデータが第2の組のデータ要素を含み、前記アンパックが前記第1の組のデータ要素の下位データ要素を前記第2の組のデータ要素の下位データ要素によってインタリーブすることを特徴とする、請求項10に記載の方法。

14. 前記第1のレジスタが64ビット長であり、前記第1のバックドデータが4個のバックドワード・データ要素を含むことを特徴とする、請求項10に記載の方法。

【発明の詳細な説明】

複合オペランドのバック演算機能を有するマイクロプロセッサ

発明の背景

発明の分野

本発明は、単一の制御信号を使用して複数のデータ要素を操作する演算を行う装置および方法を含む。本発明は、バックされたデータ（バックドデータ）・データ・タイプに対する移動演算、バック演算、およびアンパック演算の実行を可能にする。

関連技術の説明

現在、ほとんどのパーソナル・コンピュータ・システムは1つの命令によって演算を行って1つの結果を出す。命令の実行速度とプロセッサ命令の複雑さを増すことと、複数の命令を並列して実行することによって、パフォーマンスの向上が実現され、これは複雑命令セット・コンピュータ（CISC, Complex Instruction Set Computer）と呼ばれる。米国カリフォルニア州サンタクララ所在のインテル・コーポレーションが販売するIntel 80286™マイクロプロセッサのようなプロセッサがCISCプロセッサの範疇に入る。

以前のコンピュータ・システム・アーキテクチャは、CISCの概念を利用するように最適化されていた。そのようなシステムは一般に、32ビット幅のデータ・バスを持つ。しかし、コンピュータ・サポータード・コオペレーション（CSC：電子会議と混在メディア・データ操作を統合したもの）、二次元／三次元グラフィックス、画像処理、ビデオ圧縮／圧縮解除、認識アルゴリズム、および音声操作を対象にしたアプリケーションによって、パフォーマンス向上の必要が増す。しかし、命令の実行速度と複雑さを増すことが唯一の解決策である。

これらのアプリケーションの1つの一般的な点は、数ビットだけが重要な、大量のデータを操作することが多いことである。すなわち、意味のあるビットがデータ・バスのサイズよりもはるかに少ないビット数で表されるデータである。たとえば、プロセッサは8ビットおよび16ビットのデータ（たとえばビデオ画像における画素の色成分）に対して多くの演算を実行するが、それよりかなり広い

でデータ・バスとレジスタを有する。したがって、32ビットのデータ・バスとレジスタを有し、これらのアルゴリズムの1つを実行するプロセッサは、データの先頭8ビットだけが重要であるため、そのデータの処理、伝送、および記憶容量の最大75パーセントが無駄になることがある。

したがって、操作するデータを表すのに必要なビット数とプロセッサの実際のデータ伝送および記憶容量との差をより効率的に使用することによってパフォーマンスを向上させるプロセッサが望ましい。

発明の概要

複数のデータ要素に対して作用するシフト演算（シフトオペレーション）を有するプロセッサについて説明する。

このプロセッサは、第1のパックドデータを格納する第1のレジスタと、デコーダと、機能ユニットとを備える。デコーダは、制御信号入力を有する。制御信号入力は、第1の制御信号と第2の制御信号を受け取る。第1の制御信号はパック演算を示す。第2の制御信号はアンパック演算を示す。機能ユニットはデコーダとレジスタとに結合されている。機能ユニットは、第1のパックドデータを使用してパック演算とアンパック演算を行う。プロセッサは移動演算もサポートする。

本説明および図には多くの詳細が含まれるが、本発明は請求の範囲によって定義される。本発明には、それらの請求の範囲に記載されている限定だけが適用される。

図面の簡単な説明

本発明を、図面に限定的なものではなく例として図示する。同様の参照符号は同様の要素を示す。

第1図は、本発明の方法および装置を使用するコンピュータ・システムの実施例を示す図である。

第2図は、本発明のプロセッサの実施例を示す図である。

第3図は、プロセッサがレジスタ・ファイル内のデータを操作するために使用

する一般的なステップを示す流れ図である。

第4 a 図は、記憶データ・タイプを示す図である。

第4 b 図、第4 c 図、および第4 d 図は、レジスタ内整数データ表現を示す図である。

第5 a 図は、パックドデータ・タイプを示す図である。

第5 b 図、第5 c 図、および第5 d 図は、レジスタ内パックドデータ表現を示す図である。

第6 a 図は、パックドデータの使用を示す、コンピュータ・システムで使用される制御信号形式の図である。

第6 b 図は、パックドデータまたは整数データの使用を示す、コンピュータ・システムで使用可能な第2の制御信号形式の図である。

第7 図は、パックドデータに対してシフト演算を行うときにプロセッサが従う方法の一実施例を示す図である。

第8 a 図は、パックドバイト・データに対するパック演算を実施することができる回路を示す図である。

第8 b 図は、パックドワード・データに対するパック演算を実施することができる回路を示す図である。

第9 図は、パックドデータに対するアンパック演算を行う場合にプロセッサがたどる方法の一実施例を示す図である。

第10 図は、パックドデータに対するアンパック演算を実施することができる回路を示す図である。

好ましい実施例の説明

本発明の一実施例の概要

複数のデータ要素に対して作用する移動演算（ムーブオペレーション）とパック演算とアンパック演算の機能を有するプロセッサについて説明する。以下の説明では、本発明を十分に理解することができるように、回路などの多くの特定の詳細を記載する。他の場合には、本発明が無用に不明瞭にならないように、周知

の構造および技法については詳細に示さない。

定義

本発明の実施態様の説明を理解する基礎となるように、以下のように定義を示す。

ビットXないしビットY：二進数のサブフィールドを規定する。たとえば、（基数2で示した）バイト 00111010_2 のビット6ないしビット0は、1サブフィールド 11010_2 を表す。二進数の後の下付け文字「2」は基数2を示す。したがって、 1000_2 は 8_{10} に等しく、 F_{16} は 15_{10} に等しい。

R_i ：レジスタである。レジスタは、データの記憶と供給を行うことができる任意の素子である。レジスタの他の機能については後述する。レジスタはプロセッサのパッケージの一部であるとは限らない。

DEST：データ・アドレスである。

SRC1：データ・アドレスである。

SRC2：データ・アドレスである。

結果(Result)：DESTによってアドレス指定されたレジスタに格納されるデータである。

ソース1：SRC1によってアドレス指定されたレジスタに格納されているデータである。

ソース2：SRC2によってアドレス指定されたレジスタに格納されているデータである。

コンピュータ・システム

第1図を参照すると、本発明の実施例を実施することができるコンピュータ・システムが、コンピュータ・システム100として図示されている。コンピュータ・システム100は、情報を伝送するバス101またはその他の通信ハードウェアおよびソフトウェアと、バス101に結合された情報を処理するプロセッサ109とを備える。コンピュータ・システム100はさらに、バス101に結合され、プロセッサ109によって実行される情報と命令を記憶するランダム・ア

クセス・メモリ（RAM）またはその他のダイナミック記憶装置（メイン・メモリ104と呼ぶ）を備える。メイン・メモリ104は、プロセッサ109による命令の実行中に変数またはその他の中間情報を一時的に記憶するためにも使用する。

ることができる。コンピュータ・システム100は、バス101に結合され、プロセッサ109のための静的情報および命令を記憶する読取り専用メモリ（ROM）106またはその他のスタティック記憶装置あるいはその両方も備える。バス101には情報と命令を記憶するデータ記憶装置107が結合されている。

さらに、コンピュータ・システム100には磁気ディスクや光ディスクなどのデータ記憶装置107とそれに対応するディスク・ドライブを結合することができる。コンピュータ・システム100は、コンピュータ・ユーザに情報を表示するためにバス101を介して表示装置121にも結合することができる。表示装置121は、フレーム・バッファ、専用グラフィックス・レンダリング装置、陰極線管（CRT）、フラット・パネル・ディスプレイを含むことができる。プロセッサ109に情報とコマンド選択を伝えるために、英数字およびその他のキーを備える英数字入力装置122が、典型的にはバス101に結合されている。他のタイプのユーザ入力装置は、プロセッサ109に指示情報とコマンド選択を伝え、表示装置121上のカーソル移動を制御する、マウス、トラックボール、ペン、タッチ画面、カーソル指示キーなどのカーソル制御装置123である。この入力装置は一般に、第1の軸（たとえばx）と第2の軸（たとえばy）の2つの軸に2つの自由度を持ち、それによってこの装置は平面内の位置を指定することができる。しかし、本発明は、2つの自由度しかもたない入力装置には限定されない。

バス101に結合することができる他の装置は、命令、データ、またはその他の情報を、紙、フィルム、または同様のタイプの媒体などの媒体に印刷するために使用することができるハード・コピー装置124である。さらに、コンピュータ・システム100は、情報を記録するためにマイクロホンに結合されたオーディオ・ディジタイザなど、録音または再生あるいはその両方のための装置125に結合することができる。さらに、この装置は、デジタル化された音声を再生するためにデジタル-アナログ（D/A）変換器に結合されたスピーカも含むことができる。

また、コンピュータ・システム100は、コンピュータ・ネットワーク（たと

えばLAN)内の端末とすることもできる。その場合、コンピュータ・システム100は、いくつかのネットワーク化された装置を含むコンピュータ・システムのコンピュータ・サブシステムとなる。コンピュータ・システム100は、任意選択としてビデオ・ディジタイジング装置126を備える。ビデオ・ディジタイジング装置126を使用して、ビデオ画像をキャプチャし、それをコンピュータ・ネットワーク上の他の装置に送信することができる。

コンピュータ・システム100は、コンピュータ・サポーテッド・コオペレーション(CSC:電子会議と混在媒体データ操作とが統合されたもの)、二次元/三次元グラフィックス、画像処理、ビデオ圧縮/圧縮解除、認識アルゴリズム、および音声操作に対応するのに有用である。

プロセッサ

第2図にプロセッサ109の詳細図を示す。プロセッサ109は、BiCMOS、CMOS、NMOSなどのいくつかの処理技法のいずれかを使用して、1つまたは複数の基板上に実装することができる。

プロセッサ109は、プロセッサ109が使用する制御信号とデータをデコードするデコーダ202を備える。その場合、データは内部バス205を介してレジスタ・ファイル204に格納することができる。明確に言えば、実施例のレジスタは特定のタイプの回路にのみに限定されることを意味しない。むしろ実施例のレジスタは、データの記憶および供給と、本明細書に記載の機能を実行することができる。

データは、データのタイプに応じて、整数レジスタ201、レジスタ209、状態レジスタ208、または命令ポインタ・レジスタ211に格納することができる。たとえば浮動小数点レジスタなど他のレジスタをレジスタ・ファイル204に含めることができる。一実施例では、整数レジスタ201には、32ビットの整数データが格納される。一実施例では、レジスタ209にはR₀212a~R₇212hの8個のレジスタが含まれる。レジスタ209内の各レジスタ長は

64ビットである。R₀212a、R₁212b、およびR₂212cがレジスタ209内の個々のレジスタの例である。レジスタ209内の32ビットのレジス

タを、整数レジスタ201内の整数レジスタに移動させることができる。同様に、整数レジスタ内の値をレジスタ209内の32ビットのレジスタに移動させることができる。

状態レジスタ208は、プロセッサ109の状況を示す。命令ポインタ・レジスタ211には、次に実行される命令のアドレスが格納される。整数レジスタ201、レジスタ209、状態レジスタ208、および命令ポインタ・レジスタ211はすべて内部バス205に接続されている。内部バスには任意の追加のレジスタが接続される。

他の実施例では、これらのレジスタのうちのいくつかは2つの異なるタイプのデータに使用することができる。たとえば、レジスタ209と整数レジスタ201を組み合わせ、各レジスタに整数データまたはパックドデータを格納することができる。他の実施例では、レジスタ209を浮動小数点レジスタとして使用することができる。この実施例では、パックドデータはレジスタ209または浮動小数点データに格納することができる。一実施例では、組み合わせられたレジスタの長さは64ビットで、整数は64ビットで表される。この実施例では、パックドデータと整数データを格納する際に、レジスタはその2つのデータ・タイプを区別する必要がない。

機能ユニット203は、プロセッサ109が行う演算（オペレーション）を実行する。このような演算には、シフト、加算、減算、乗算などが含まれる。機能ユニット203は内部バス205に接続している。キャッシュ206は、プロセッサ109の任意選択要素であり、たとえばメイン・メモリ104からのデータまたは制御信号あるいはその両方をキャッシュするために使用される。キャッシュ206は、デコーダ202に接続され、制御信号207を受信するように接続されている。

第3図に、プロセッサ109の動作概要を示す。すなわち、第3図にはプロセッサ109がパックドデータに対する演算、アンパックドデータに対する演算、または他の何らかの操作を実行する間にたどるステップが示されている。たとえ

ば、このような操作にはレジスタ・ファイル204内のレジスタに、キャッシュ

206、メイン・メモリ104、読取り専用メモリ（ROM）106、またはデータ記憶装置107からデータをロードする操作が含まれる。本発明の一実施例では、プロセッサ109は、米国カリフォルニア州サンタクララ所在のインテル・コーポレーションが販売するIntel 80486™によってサポートされる命令のほとんどをサポートする。本発明の他の実施例では、プロセッサ109は米国カリフォルニア州サンタクララ所在のインテル・コーポレーションが販売するIntel 80486™によってサポートされるすべての演算をサポートする。本発明の他の実施例では、プロセッサ109は、すべて米国カリフォルニア州サンタクララ所在のインテル・コーポレーションが販売するPentium™プロセッサ、Intel 80486™プロセッサ、80386™プロセッサ、Intel 80286™プロセッサ、およびIntel 8086™プロセッサによってサポートされるすべての演算をサポートする。本発明の他の実施例では、プロセッサ109は、米国カリフォルニア州サンタクララ所在のインテル・コーポレーションが定義するIA™（インテル・アーキテクチャ）でサポートされるすべての演算をサポートする（米国カリフォルニア州サンタクララのインテルから入手可能な「Microprocessors, Intel Data Books volume 1およびvolume 2、1992年および1993年刊」を参照）。一般に、プロセッサ109はPentium™プロセッサの現行命令セットをサポートすることができるが、将来の命令と本明細書に記載の命令を組み込むように修正することもできる。重要なのは、汎用プロセッサ109が、本明細書に記載の演算に加えて、従来使用されていた演算をサポートすることができることである。

ステップ301で、デコーダ202がキャッシュ206またはバス101から制御信号207を受け取る。デコーダ202は、制御信号をデコードして、実行すべき演算を判断する。

ステップ302で、デコーダ202はレジスタ・ファイル204またはメモリ内の記憶場所にアクセスする。制御信号207で指定されたレジスタ・アドレスに応じて、レジスタ・ファイル204内のレジスタかメモリ内の記憶場所のどち

らかにアクセスする。たとえば、バックドデータに対する演算の場合、制御信号207にはSRC1、SRC2、およびDESTレジスタ・アドレスを含めることができる。SRC1は第1のソース・レジスタのアドレスである。SRC2は第2のソース・レジスタのアドレスである。すべての演算が2つのソース・アドレスを必要とするわけではないので、場合によってはSRC2アドレスは任意選択である。SRC2アドレスが不要な場合、SRC1アドレスのみが使用される。DESTは、結果データが格納される宛先レジスタのアドレスである。一実施例では、SRC1またはSRC2はDESTとしても使用される。SRC1、SRC2、およびDESTについては第6a図および第6b図を参照しながら詳述する。対応するレジスタに格納されているデータをそれぞれSource1、Source2、およびResultと呼ぶ。これらの各データの長さは64ビットである。

本発明の他の実施例では、SRC1、SRC2、およびDESTのいずれか1つまたは全部は、プロセッサ109のアドレス可能記憶空間内の記憶場所を規定することができる。たとえば、SRC1はメイン・メモリ104内の記憶場所を識別し、SRC2は整数レジスタ201内の第1のレジスタを識別し、DESTはレジスタ209内の第2のレジスタを識別する。本明細書では説明を簡単にするために、レジスタ・ファイル204へのアクセスについて言及するが、これらのアクセスはレジスタ・ファイル204の代わりにメモリに対して行うこともできる。

本発明の他の実施例では、命令コードはSRC1とSRC2の2つのアドレスしか含まない。この実施例では、演算の結果はSRC1レジスタまたはSRC2レジスタに格納される。すなわち、SRC1（またはSRC2）をDESTとして使用する。このタイプのアドレス指定は、2つのアドレスしか持たない以前のCISC命令に対応する。これによって、デコーダ202における複雑さが減少する。この実施例では、SRC1レジスタに含まれるデータを破壊してはならない場合、演算を実行する前にそのデータをまず別のレジスタにコピーしなければならないことに留意されたい。コピーには追加の命令が必要になる。本明細書では説明を簡単にするために、3アドレスのアドレス指定方式について説明する

(すなわちSRC1、SRC2、およびDEST)。しかし、一実施例では制御信号にSRC1とSRC2しか含めることができず、SRC1（またはSRC2）によって宛先レジスタを識別することを想起されたい。

制御信号が演算を要求とする場合、ステップ303で、機能ユニット203がレジスタ・ファイル204内のアクセス・データに対してその演算を実行するように使用可能にされる。機能ユニット203で演算が実行されると、ステップ304でその結果が制御信号207の要件に従ってレジスタ・ファイル204に戻されて格納される。

データ形式および記憶形式

第4a図に、第1図のコンピュータ・システムで使用可能なデータ形式をいくつか示す。これらのデータ形式は固定小数点である。プロセッサ109はこれらのデータ形式を操作することができる。マルチメディア・アルゴリズムはこれらのデータ形式を使用することが多い。バイト401は8ビットの情報を含む。ワード402は16ビットの情報、すなわち2バイトを含む。ダブルワード403は32ビットの情報、すなわち4バイトを含む。したがって、プロセッサ109はこれらの記憶データ形式のうちの任意の1つに対して操作を行うことができる制御信号を実行する。

以下の説明では、ビット、バイト、ワード、およびダブルワード・サブフィールドについて言及する。たとえば、(基数2で示す) バイト00111010₂のビット6ないしビット0はサブフィールド111010₂を表す。

第4b図ないし第4d図に、本発明の一実施例で使用するレジスタ内表現を示す。たとえば、無符号バイトのレジスタ内表現410によって、整数レジスタ201内のレジスタに格納されているデータを表すことができる。一実施例では、整数レジスタ201内のレジスタ長は64ビットである。他の実施例では、整数レジスタ201内のレジスタ長は32ビットである。説明を簡単にするために、以下の説明では64ビットの整数レジスタについて説明するが、32ビットの整数レジスタを使用することもできる。

無符号バイトのレジスタ内表現410は、プロセッサ109が整数レジスタ2

01にバイト401を格納し、そのレジスタ内のビット7ないしビット0の先頭8ビットがそのデータ・バイト401専用であることを示している。これらのビットを{b}と示す。このバイトを正しく表すには、残りの56ビットがゼロでなければならない。符号付きバイトのレジスタ内表現411の場合、整数レジスタ201にはデータはビット6ないしビット0の先頭7ビットにデータとして格納される。7番目のビットは符号ビットを表し、{s}で示す。残りのビット63ないしビット8はそのバイトの符号の継続である。

無符号ワードのレジスタ内表現412は、レジスタ201のうちの1つのレジスタに格納される。ビット15ないしビット0には、無符号ワード402が入れる。これらのビットを{w}で示す。このワードを正しく表すには、残りのビット63ないしビット16はゼロでなければならない。符号付きワード402は、符号付きワードのレジスタ内表現413が示すように、ビット14ないしビット0に格納される。残りのビット63ないしビット15は符号フィールドである。

ダブルワード403は、無符号ダブルワードのレジスタ内表現414または符号付きダブルワードのレジスタ内表現415として格納することができる。無符号ダブルワードのレジスタ内表現414のビット31ないしビット0がデータである。これらのビットを{d}で示す。この無符号ダブルワードを正しく表すには、残りのビット63ないしビット32はゼロでなければならない。整数レジスタ201には、符号付きダブルワードのレジスタ内表現415が、そのビット30ないしビット0に格納される。残りのビット63ないしビット31は符号フィールドである。

前述の第4b図ないし第4d図に示すように、データ・タイプによっては64ビット幅のレジスタに格納するのは非効率的な格納方法である。たとえば、無符号バイトのレジスタ内表現410を格納する場合、ビット63ないしビット8はゼロでなければならず、ビット7ないしビット0にしか非ゼロ・ビットを入れることができない。したがって、64ビット・レジスタに1バイトを格納するプロセッサは、レジスタの容量の12.5%しか使用しない。同様に、機能ユニット203によって実行される命令は先頭の数ビットしか重要ではない。

第5 a 図に、パックされたデータすなわちパックドデータのデータ形式を示す。各パックドデータは複数の独立したデータ要素を含む。パックドバイト（パックされたバイト）501、パックドワード（パックされたワード）502、パックドダブルワード（パックされたダブルワード）503の3つのパックドデータ形式が図示されている。パックドバイトは、本発明の一実施例では64ビット長であり、8個のデータ要素を含む。各データ要素は1バイト長である。一般に、データ要素は1つのレジスタ（または記憶場所）に同じ長さの他のデータ要素と共に格納される個々のデータである。本発明の一実施例では、1つのレジスタに格納されるデータ要素の数は、64ビットをデータ要素のビット長で割った商である。

パックドワード502は64ビット長であり、4個のワード402データ要素を含む。各ワード402データ要素は、16ビットの情報を含む。

パックドダブルワード503は64ビット長であり、2個のダブルワード403データ要素を含む。各ダブルワード403データ要素は32ビットの情報を含む。

第5 b 図ないし第5 d 図にレジスタ内パックドデータ記憶表現を示す。無符号パックドバイトのレジスタ内表現510は、レジスタR₀212a～R₀212afのうちの1つにパックドバイト501が格納されている様子を示している。各バイト・データ要素の情報は、バイト0はビット7ないしビット0に格納され、バイト1はビット15ないしビット8、バイト2はビット23ないしビット16、バイト3はビット31ないしビット24、バイト4はビット39ないしビット32、バイト5はビット47ないしビット40、バイト6はビット55ないしビット48、バイト7はビット63ないしビット56に格納される。したがって、レジスタ内ですべての使用可能ビットが使用される。この記憶構成によって、プロセッサの記憶効率が向上する。また、8個のデータ要素にアクセスして、1つの操作を8個のデータ要素に同時に実行することができるようになる。符号付きパックドバイトのレジスタ内表現511も同様にレジスタ209内のレジスタに格納される。どのバイト・データ要素でも8番目のビットのみが必要な符号ビットであり、他のビットは使用してもしなくても符号が示されることに留意された

い。

無符号パックドワードのレジスタ内表現512は、ワード3ないしワード0がレジスタ209のうちの1つのレジスタに格納される様子を示している。ビット15ないしビット0にはワード0のデータ要素情報が入り、ビット31ないしビット16にはデータ要素ワード1の情報が入り、ビット47ないしビット32にはデータ要素ワード2の情報が入り、ビット63ないしビット48にはデータ要素ワード3の情報が入る。符号付きパックドワードのレジスタ内表現513は無符号パックドワードのレジスタ内表現512と同様である。各ワード・データ要素の16番目のビットにのみ、必要な符号インジケータが入ることに留意されたい。

無符号パックドダブルワードのレジスタ内表現514は、レジスタ209に2個のダブルワード・データ要素が格納される様子を示している。ダブルワード0はレジスタのビット31ないしビット0に格納される。ダブルワード1はレジスタのビット63ないしビット32に格納される。符号付きパックドダブルワードのレジスタ内表現515は無符号パックドダブルワードのレジスタ内表現514と同様である。必要な符号ビットはダブルワード・データ要素の32番目のビットであることに留意されたい。

前述のように、レジスタ209はパックドデータと整数データの両方に使用することができる。本発明のこの実施例では、アドレス指定されたレジスタ、たとえばR_{212a}にパックドデータと単純整数／固定小数点データのどちらが格納されているかを追跡するために、個々のプログラミング・プロセッサ109が必要である。他の実施例では、プロセッサ109はレジスタ209の個々のレジスタに格納されているデータのタイプを追跡することができる。この代替実施例では、たとえば単純／固定小数点整数データに対してパック加算を行おうとした場合、エラーを生成することができる。

制御信号の形式

以下に、プロセッサ109がパックドデータを操作するために使用する制御信号形式の一実施例について説明する。本発明の一実施例では、制御信号は32ビ

ットで表される。デコーダ202はバス101から制御信号207を受け取る
とができる。他の実施例では、デコーダ202はキャッシュ206からもそのよ
うな制御信号を受け取ることができる。

第6a図にパックドデータを操作する制御信号の一般的な形式を示す。命令フ
ィールドOP601（ビット31ないしビット26）は、たとえば、パック加算
、パック減算など、プロセッサ109によって実行される演算に関する情報を与
える。SRC1 602（ビット25ないしビット20）は、レジスタ209内
のレジスタのソース・レジスタ・アドレスを供給する。このソース・レジスタは
、制御信号の実行で使用される第1のパックドデータSource1を保持する
。同様に、SRC2 603（ビット19ないしビット14）には、レジスタ2
09内のレジスタのアドレスが入れられる。この第2のソース・レジスタは、演
算の実行時に使用されるパックドデータSource2を保持する。DEST6
05（ビット5ないしビット0）にはレジスタ209内のレジスタのアドレスが
入れられる。この宛先レジスタには、パックドデータ演算の結果パックドデータ
Resultが格納される。

制御ビットSZ610（ビット12およびビット13）は、第1および第2の
パックドデータ・ソース・レジスタ内のデータ要素の長さを示す。SZ610が
0₁に等しい場合、パックドデータはパックドバイト501としてフォーマッ
トされる。SZ610が1₀に等しい場合、パックドデータはパックドワード
502としてフォーマットされる。しかし、0₀または1₁と等しいSZ61
0を受け取った場合、他の実施例では、これらの値のうちの1つを使用してパッ
クドダブルワード503を示すことができる。

制御ビットT611（ビット11）は、演算を飽和モードで行うかどうかを示
す。T611が1の場合、飽和演算が行われる。T611がゼロの場合、非飽和
演算が行われる。飽和演算については後述する。

制御ビットS612（ビット10）は、符号付き演算の使用を示す。S612
が1の場合、符号付き演算が行われる。S612がゼロの場合、無符号演算が行
われる。

第6b図に、バックドデータを操作する制御信号の第2の一般的形式を示す。
この形式は、米国イリノイ州マウント・プロスペクトP. O. Box 7641イ

ンテル・コーポレイションのLiterature Salesから入手可能な
"Pentium™ Processor Family User's Manual"に記載されている汎用整数命令コード形式に対応する。OP601、
SZ610、T611、およびS612がすべて組み合わせられて1つの大きなフ
ィールドになることに留意されたい。制御信号によっては、ビット3ないし5が
SRC1 602となる。一実施例では、SRC1 602アドレスがある場合
、ビット3ないし5はDEST605にも対応する。SRC2 603アドレス
が存在する代替実施例ではビット0ないし2もDEST605に対応する。パッ
クドシフト即値演算のような他の制御信号の場合、ビット3ないし5は命令コー
ド・フィールドの拡張部を表す。一実施例では、この拡張部によってプログラマ
はシフト・カウント値などの即値を制御信号と共に組み込むことができる。一実
施例では、即値は制御信号の後に続く。これについては"Pentium™ P
rocessor Family User's Manual"の付録FのF
-1~F-3ページに詳述されている。ビット0ないし2はSRC2 603を
表す。この汎用形式によって、レジスタからレジスタ、メモリからレジスタ、メ
モリによるレジスタ、レジスタによるレジスタ、即値によるレジスタ、レジスタ
からメモリのアドレス指定を行うことができる。また、一実施例では、この汎用
形式は整数レジスタからレジスタと、レジスタから整数レジスタへのアドレス指
定もサポートする。

飽和／非飽和の説明

前述のように、T611は演算が任意選択で飽和するかどうかを示す。飽和を
可能にした演算の結果がデータの範囲からオーバーフローまたはアンダーフロー
する場合、その結果はクランプされる。クランプとは、結果がその範囲の最大値
または最小値を超える場合、その結果を最大値または最小値に設定することを意
味する。アンダーフローの場合、飽和によって結果がその範囲内の最低値にクラ
ンプされ、オーバーフローの場合は最高値にクランプされる。各データ形式の許

容範囲を表1に示す。

表1

データ形式	最小値	最大値
無符号バイト	0	255
符号付きバイト	-128	127
無符号ワード	0	65535
符号付きワード	-32768	32767
無符号ダブルワード	0	$2^{64} - 1$
符号付きダブルワード	-2^{63}	$2^{63} - 1$

前述のように、T611は飽和演算を行うかどうかを示す。したがって、無符号バイト・データ形式を使用し、演算結果＝258で、飽和を使用可能にしていた場合、結果は演算の宛先レジスタに格納される前に255にクランプされることになる。同様に、演算結果＝-32999で、プロセッサ109が飽和を使用可能にして符号付きデータ形式を使用した場合、結果は演算の宛先レジスタに格納される前に-32768にクランプされることになる。

データ操作演算

本発明の一実施例では、標準CISC命令セット（アンパックドデータ演算）をサポートするだけでなくパックドデータのシフト演算もサポートすることによって、マルチメディア・アプリケーションのパフォーマンスを向上させる。このようなパックドデータ演算には、加算、減算、乗算、比較、シフト、AND、およびXORを含めることができる。しかし、これらの演算を十分に利用するためには、データ操作演算を組み込む必要があると判断されている。このようなデータ操作演算には、移動、パック、アンパックを含めることができる。移動、パック、およびアンパックは、プログラマにとってより使いやすい形式のパックドデ

ータを生成することによって他の演算の実行を容易にする。

他のバック演算の詳細な背景については、年 月 日出願の特許出願第 号“A Microprocessor Having a Compare Operation”、年 月 日出願の特許出願第 号“A Microprocessor Having a Multiply Operation”、年 月 日出願の特許出願第 号“A Novel Processor Having Shift Operations”、1993年12月30日出願の特許出願第08/176123号“A Method and Apparatus Using Packed Data in a Processor”、および1993年12月30日出願の特許出願第08/175772号“A Method and Apparatus Using Novel Operations in a Processor”を参照されたい。これらはすべて本発明の譲渡人に譲渡される。

移動演算

移動演算は、レジスタ209との間でデータを送受信する。一実施例では、SRC2 603がソース・データを含むアドレスであり、DEST605がデータの送り先のアドレスである。この実施例では、SRC1 602は使用しない。他の実施例では、SRC1 602はDEST605である。

移動演算の説明のために、レジスタと記憶場所との区別を設ける。レジスタはレジスタ・ファイル204内にあり、記憶場所はたとえばキャッシュ206、メイン・メモリ104、ROM106、データ記憶装置107内とすることができる。

移動演算によって、データを記憶場所からレジスタ209、レジスタ209から記憶場所、レジスタ209内のレジスタからレジスタ209内の第2のレジスタに移動することができる。一実施例では、パックドデータは整数データを記憶するために使用されるレジスタとは異なるレジスタに格納される。この実施例では、移動演算によって整数レジスタ201からレジスタ209にデータを移動す

ることができる。たとえば、プロセッサ109において、レジスタ209にパックドデータが格納され、整数レジスタ201に整数データが格納されている場合、移動命令を使用して整数レジスタ201からレジスタ209にデータを移動したり、その逆を行ったりすることができる。

一実施例では、移動のためにメモリ・アドレスを示すと、記憶場所（最下位バイトを示す記憶場所）にある8バイトのデータがレジスタ209内のレジスタにロードされるかまたはそのレジスタから格納される。レジスタ209内のレジスタが示された場合、そのレジスタの内容がレジスタ209内の第2のレジスタに移動されるかまたはそこからロードされる。整数レジスタ201が64ビット長であって、整数レジスタを指定した場合、そのレジスタ内の8バイトのデータがレジスタ209内のレジスタにロードされるかまたはそのレジスタから格納される。

一実施例では、整数は32ビットで表される。レジスタ209から整数レジスタ201への移動演算を行う場合、パックドデータの下位32ビットだけが指定された整数レジスタに移動される。一実施例では、上位32ビットはゼロにされる。同様に、整数レジスタ201からレジスタ209への移動を実行すると、レジスタ209内のレジスタの下位32ビットだけがロードされる。一実施例では、プロセッサ109はレジスタ209内のレジスタとメモリとの間の32ビットの移動演算をサポートする。他の実施例では、パックドデータの上位32ビットに対して32ビットのみの移動が行われる。

パック演算

本発明の一実施例では、SRC1 602レジスタにデータ（Source1）が入れられ、SRC2 603レジスタにデータ（Source2）が入れられ、DEST605レジスタには演算の結果データ（Result）が入れられる。すなわち、Source1の部分とSource2の部分が一緒にパックされて結果が生成される。

一実施例では、パック演算は、ソース・パックドワード（またはダブルワード）の下位バイト（またはワード）をResultのバイト（またはワード）にパッ

クすることによって、パックドワード（またはダブルワード）がパックドバイト（またはワード）に変換する。一実施例では、パック演算によって、クワード・パックドワードがパックドダブルワードに変換される。この演算は、任意選択により符号付きデータを使用して行うことができる。さらにこの演算は、任意選択により飽和を使用して行うことができる。

第7図に、パックドデータに対してパック演算を行う方法の一実施例を示す。この実施例は、第2図のプロセッサ109で実施することができる。

ステップ701で、プロセッサ109が受け取った制御信号207をデコーダ202がデコードする。したがって、デコーダ202は、適切なシフト演算の命令コードと、レジスタ209内のSRC1 602、SRC2 603、およびDEST605アドレスと、飽和／非飽和と、符号付き／無符号と、パックドデータ内のデータ要素の長さとをデコードする。前述のように、SRC1 602（またはSRC2 603）をDEST605として使用することができる。

ステップ702で、SRC1 602アドレスとSRC2 603アドレスが与えられた場合、内部バス205を介してデコーダ202がレジスタ・ファイル204内のレジスタ209にアクセスする。レジスタ209は機能ユニット203にSRC1 602レジスタに格納されているパックドデータ（Source 1）と、SRC2 603レジスタに格納されているパックドデータ（Source 2）を供給する。すなわち、レジスタ209は、パックドデータを内部バス205を介して機能ユニット203に伝達する。

ステップ703で、デコーダ202は機能ユニット203が適切なパック演算を実行することができるようにする。デコーダ202は、さらに、内部バス205を介して飽和とSource 1内およびSource 2内のデータ要素のサイズも伝達する。任意選択で飽和を使用して、結果データ要素内のデータの値を最大化する。Source 1またはSource 2内のデータ要素の値が、Resultのデータ要素が表すことができる値の範囲より大きいまたは小さい場合、それに対応する結果データ要素がその最高値または最低値に設定される。たとえば、Source 1およびSource 2のワード・データ要素内の符号付き値が0×80（またはダブルワードの場合は0×8000）よりも小さい場合、

結

果のバイト（またはワード）データ要素が 0×80 （またはダブルワードの場合は 0×8000 ）にクランプされる。Source 1およびSource 2のワード・データ要素内の符号付き値が $0 \times 7F$ （またはダブルワードの場合は $0 \times 7FFF$ ）よりも大きい場合は、結果のバイト（またはワード）データ要素が $0 \times 7F$ （または $0 \times 7FFF$ ）にクランプされる。

ステップ710で、データ要素のサイズによって次にどのステップを実行するかが決まる。データ要素のサイズが16ビット（バックワード502データ）の場合、機能ユニット203はステップ712を実行する。しかしバックデータ内のデータ要素のサイズが32ビット（バックダブルワード503データ）の場合、機能ユニット203はステップ714を実行する。

ソース・データ要素のサイズが16ビットであると仮定すると、ステップ712が実行される。ステップ712では、以下のように行う。Source 1のビット7～0はResultのビット7～0である。Source 1のビット23～16はResultのビット15～8である。Source 1のビット39～32はResultのビット23～16である。Source 1のビット63～56はResultのビット31～24である。Source 2のビット7～0はResultのビット39～32である。Source 2のビット23～16はResultのビット47～40である。Source 2のビット39～32はResultのビット55～48である。Source 2のビット63～56はResultのビット31～24である。飽和を設定した場合は、Resultデータ要素をクランプすべきかどうかを判断するために各ワードの上位ビットが検査される。

ソース・データ要素のサイズが32ビットであると仮定すると、ステップ714が実行される。ステップ714では、以下のように行われる。Source 1のビット15～0はResultのビット15～0である。Source 1のビット47～32はResultのビット31～16である。Source 2のビット15～0はResultのビット47～32である。Source 2のビット

ト47～32はResultのビット63～48である。飽和を設定した場合は、Resultデータ要素をクランプすべきかどうかを判断するために各ダブルワ

ードの上位ビットが検査される。

一実施例では、ステップ712のバック演算が同時に行われる。しかし、他の実施例では、このバック演算は順次に行われる。他の実施例では、このバック演算の一部が同時に行われ、一部は順次に行われる。これは、ステップ714のバック演算にも適用される。

ステップ720で、ResultがDEST605レジスタに格納される。

表2に、飽和を使用しないパック無符号ワード演算のレジスタ内表現を示す。最初の行のビットはSource1のパックドデータ表現である。2番目の行のビットはSource2のデータ表現である。3番目の行のビットはResultのパックドデータ表現である。各データ要素ビットの下に数字はデータ要素番号である。たとえば、Source1データ要素3は10000000である。

表 2

Source1							
00101010	01010101	01010101	11111111	10000000	01110000	10001111	10001000
3		2		1		0	

Source2							
00000000	00000000	11000000	00000000	11110011	00000000	10001110	10001000
3		2		1		0	

Result							
00000000	00000000	00000000	10001000	01010101	11111111	01110000	10001000
7	6	5	4	3	2	1	0

表3に、飽和を使用したパック符号ダブルワード演算のレジスタ内表現を示す。

表 3

Source1							
00101010 01010101 01010101 11111111				10000000 01110000 10001111 10001000			
1				0			
Source2							
00000000 00000000 11000000 00000000				11110011 00000000 10001110 10001000			
1				0			
Result							
11000000 00000000		10000000 00000000		01111111 11111111		10000000 00000000	
3		2		1		0	

バック回路

本発明の一実施例では、バック演算の効率的な実行を実現するために並列処理を使用する。第 8 a 図および第 8 b 図に、パックドデータに対するバック演算を行う回路の一実施例を示す。

第 8 a 図および第 8 b 図の回路は、演算制御回路 8 0 0 と、結果レジスタ 8 5 2 と、結果レジスタ 8 5 3 と、8 個の 1 6 ビットから 8 ビットの飽和検査回路と、4 個の 3 2 ビットから 1 6 ビットの飽和検査回路を備える。

演算制御回路 8 0 0 は、デコーダ 2 0 2 から情報を受け取ってバック演算を可能にする。演算制御回路 8 0 0 は飽和値を使用して各飽和検査回路の飽和検査を可能にする。ソース・パックドデータのサイズがワード・パックドデータ 5 0 3 の場合、演算制御回路 8 0 0 によって出力イネーブル 8 3 1 が設定される。これによって出力レジスタ 8 5 2 の出力がイネーブルにされる。ソース・パックドデータのサイズがダブルワード・パックドデータ 5 0 4 の場合、演算制御回路 8 0 0 によって出力イネーブル 8 3 2 が設定される。これによって、出力レジスタ 8 5 3 の出力がイネーブルにされる。

各飽和検査回路は飽和を選択的に検査することができる。飽和の検査を使用不能にした場合、各飽和検査回路はただ下位ビットを通過させて結果レジスタ内の

対応する位置に入れるだけである。飽和の検査を使用可能にした場合、各飽和検査回路は上位ビットを検査して結果をクランプすべきかどうかを判断する。

飽和検査回路 8 1 0 ないし 8 1 7 は 1 6 ビットの入力と 8 ビットの出力を有す

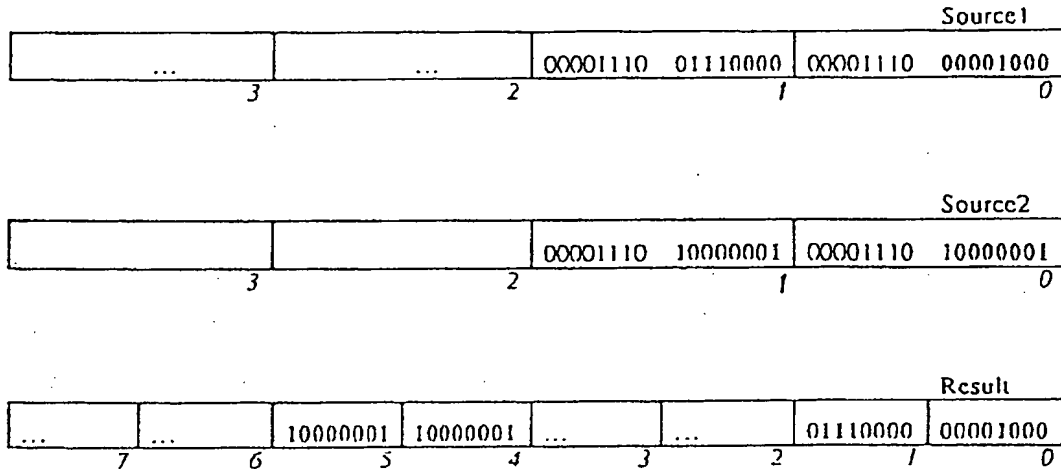
る。8ビットの出力は入力の下位8ビットか、または任意選択によりクランプ値（ 0×80 、 $0 \times 7F$ 、または $0 \times FF$ ）である。飽和検査回路810は、Source1のビット15ないし0を受け取り、結果レジスタ852に対するビット7ないし0を出力する。飽和検査回路811は、Source1のビット31ないし16を受け取り、結果レジスタ852に対するビット15ないし8を出力する。飽和検査回路812は、Source1のビット47ないし32を受け取り、結果レジスタ852に対するビット23ないし16を出力する。飽和検査回路813は、Source1のビット63ないし48を受け取り、結果レジスタ852に対するビット31ないし24を出力する。飽和検査回路814は、Source2のビット15ないし0を受け取り、結果レジスタ852に対するビット39ないし32を出力する。飽和検査回路815は、Source2のビット31ないし16を受け取り、結果レジスタ852に対するビット47ないし40を出力する。飽和検査回路816は、Source2のビット47ないし32を受け取り、結果レジスタ852に対するビット55ないし48を出力する。飽和検査回路817は、Source2のビット63ないし48を受け取り、結果レジスタ852に対するビット63ないし56を出力する。

飽和検査回路820ないし飽和検査回路823は、32ビットの入力と16ビットの出力を有する。16ビットの出力は、入力の下位16ビットか、または任意選択によりクランプ値（ 0×8000 、 $0 \times 7FFF$ 、または $0 \times FFFF$ ）である。飽和検査回路820は、Source1のビット31ないしゼロを受け取り、結果レジスタ853に対するビット15ないし0を出力する。飽和検査回路821はSource1のビット63ないし32を受け取り、結果レジスタ853に対するビット31ないし16を出力する。飽和検査回路822は、Source2のビット31ないし0を受け取り、結果レジスタ853に対するビット47ないし32を出力する。飽和検査回路823は、Source2のビット63ないし32を受け取り、結果レジスタ853に対するビット63ないし48を出力する。

たとえば、表4に飽和を使用しない無符号バックワードを示す。演算制御回

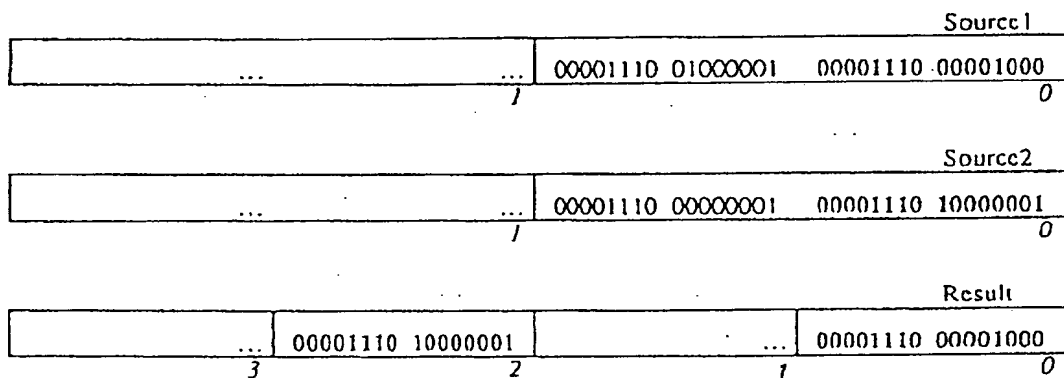
路800は、結果レジスタ852が結果[63:0]860を出力することができるようにする。

表 4



しかし、飽和なしの無符号パックドダブルワード演算を行う場合、演算制御回路800は結果レジスタ853が結果[63:0]860を出力することができるようにする。表5にこの結果を示す。

表 5



アンパック演算

一実施例では、アンパック演算によって、2つのソース・パックドデータの下位パックドバイト、ワード、またはダブルワードをインタリーブして、結果のパックドバイト、ワード、またはダブルワードを生成する。

第9図に、パックドデータに対するアンパック演算を行う方法の一実施例を示す。この実施例は、第2図のプロセッサ109において実施することができる。

先にステップ701およびステップ702が実行される。ステップ903で、デコーダ202がアンパック演算を行うように機能ユニット203を使用可能にする。デコーダ202は内部バス205を介してSource1およびSource2内のデータ要素のサイズを伝達する。

ステップ910で、このデータ要素のサイズによって次にどのステップを実行するかが決まる。データ要素のサイズが8ビット（パックドバイト501データ）の場合、機能ユニット203はステップ712を実行する。しかし、パックドデータ内のデータ要素のサイズが16ビット（パックドワード502データ）の場合、機能ユニット203はステップ714を実行する。しかし、パックドデータ内のデータ要素のサイズが32ビット（パックドダブルワード503データ）の場合、機能ユニット203はステップ716を実行する。

ソース・データ要素のサイズが8ビットであると仮定すると、ステップ712が実行される。ステップ712では以下のように行われる。Source1のビット7～0はResultのビット7～0である。Source2のビット7～0はResultのビット15～8である。Source1のビット15～8はResultのビット23～16である。Source2のビット15～8はResultのビット31～24である。Source1のビット23～16はResultのビット39～22である。Source2のビット23～16はResultのビット47～40である。Source1のビット31～24はResultのビット55～48である。Source2のビット31～24はResultのビット63～56である。

ソース・データ要素のサイズが16ビットであると仮定すると、ステップ714が実行される。ステップ714では以下のように行われる。Source1の

ビット15～0はResultのビット15～0である。Source2のビット15～0はResultのビット31～16である。Source1のビット31～16はResultのビット47～32である。Source2のビット31～16はResultのビット63～48である。

ソース・データ要素のサイズが32ビットであると仮定すると、ステップ71

6が実行される。ステップ716では以下のように行われる。Source1のビット31～0がResultのビット31～0である。Source2のビット31～0がResultのビット63～32である。

一実施例では、ステップ712のアンパック演算が同時に行われる。しかし、他の実施例では、このアンパック演算は順次に行われる。他の実施例では、このアンパック演算の一部が同時に行われ、一部は順次に行われる。これは、ステップ714および716のアンパック演算にも適用される。

ステップ720で、ResultがDEST605レジスタに格納される。

表6に、アンパック・バイト演算のレジスタ内表現を示す。

表6

Source1							
00101010	01010101	01010101	11111111	10000000	01110000	10001111	10001000
7	6	5	4	3	2	1	0

Source2							
00000000	00000000	11000000	00000000	11110011	00000000	10001110	10001000
7	6	5	4	3	2	1	0

Result							
11110011	10000000	00000000	01110000	10001110	10001111	10001000	10001000
7	6	5	4	3	2	1	0

表7に、アンパック・ワード演算のレジスタ内表現を示す。

表7

Source1							
00101010	01010101	01010101	11111111	10000000	01110000	10001111	10001000
3		2		1		0	

Source2							
00000000	00000000	11000000	00000000	11110011	00000000	10001110	10001000
3		2		1		0	

Result							
11110011	00000000	10000000	01110000	10001110	10001000	10001111	10001000
3		2		1		0	

表8に、アンパック・ダブルワード演算のレジスタ内表現を示す。

表 8

				Source1			
00101010	01010101	01010101	11111111	10000000	01110000	10001111	10001000
1				0			
				Source2			
00000000	00000000	11000000	00000000	11110011	00000000	10001110	10001000
1				0			
				Result			
11110011	00000000	10001110	10001000	10000000	01110000	10001111	10001000
1				0			

アンパック回路

本発明の一実施例では、アンパック演算の効率的な実行を実現するために並列処理を使用する。第10図に、パックドデータに対するアンパック演算を実行することができる回路の一実施例を示す。

第10図の回路は、演算制御回路800と結果レジスタ1052と結果レジス

タ1053と結果レジスタ1054とを備える。

演算制御回路800は、デコーダ202からアンパック演算をイネーブルにする情報を受け取る。ソース・パックドデータのサイズがバイト・パックドデータ502の場合、演算制御回路800によって出力イネーブル1032が設定される。これによって、結果レジスタ1052の出力がイネーブルにされる。ソース・パックドデータのサイズがワード・パックドデータ503の場合、演算制御回路800によって出力イネーブル1033が設定される。これによって出力レジスタ1053の出力がイネーブルにされる。ソース・パックドデータのサイズがダブルワード・パックドデータ504の場合、演算制御回路800によって出力イネーブル1034が設定される。これによって、出力結果レジスタ1054の出力がイネーブルにされる。

結果レジスタ1052は以下の入力を有する。Source1のビット7～0は結果レジスタ1052のビット7～0である。Source2のビット7～0は結果レジスタ1052のビット15～8である。Source1のビット15～8は結果レジスタ1052のビット23～16である。Source2のビッ

ト15～8は結果レジスタ1052のビット31～24である。Source1のビット23～16は結果レジスタ1052のビット39～32である。Source2のビット23～16は結果レジスタ1052のビット47～40である。Source1のビット31～24は結果レジスタ1052の55～48である。Source2のビット31～24は結果レジスタ1052のビット63～56である。

結果レジスタ1053は以下の入力を有する。Source1のビット15～0は結果レジスタ1053の15～0である。Source2のビット15～0は結果レジスタ1053の31～16である。Source1のビット31～16は結果レジスタ1053のビット47～32である。Source2のビット31～16は結果レジスタ853のビット63～48である。

結果レジスタ1054は以下の入力を有する。Source1のビット31～0は結果レジスタ1054のビット31～0である。Source2のビット31～0は結果レジスタ1054のビット63～32である。

たとえば、表9ではアンパック・ワード演算が実行される。演算制御回路800は結果レジスタ1053が結果[63:0]860を出力することができるようにする。

表 9

Source1					
...	...	00001110	01110000	00001110	00001000
3	2	1	0		
Source2					
...	...	00001110	00000001	00001110	10000001
3	2	1	0		
Result					
00001110	00000001	00001110	01110000	00001110	10000001
3	2	1	0		

しかし、アンパック・ダブルワード演算を行う場合、演算制御回路800は結果

レジスタ1054が結果[63:0]860を出力することができるようにする。
表10にその結果を示す。

表 1 0

Source1			
...	00001110 01000001	00001110 00001000	0
1			
Source2			
...	00001110 00000001	00001110 10000001	0
1			
Result			
00001110 00000001	00001110 10000001	00001110 01000001	00001110 00001000
1			
0			

したがって、移動演算、パック演算、およびアンパック演算によってマルチメディア・データ要素を操作することができる。従来技術のプロセッサでは、これらのタイプの操作を行うために、単一のパック移動演算、パック演算、またはアンパック演算を行うために複数の別々の演算を実行する必要がある。一実施例では、パックドデータ演算のためのデータ線はすべて関係のあるデータを伝送する。これによって、コンピュータ・システムのパフォーマンスが向上する。

【図 1】

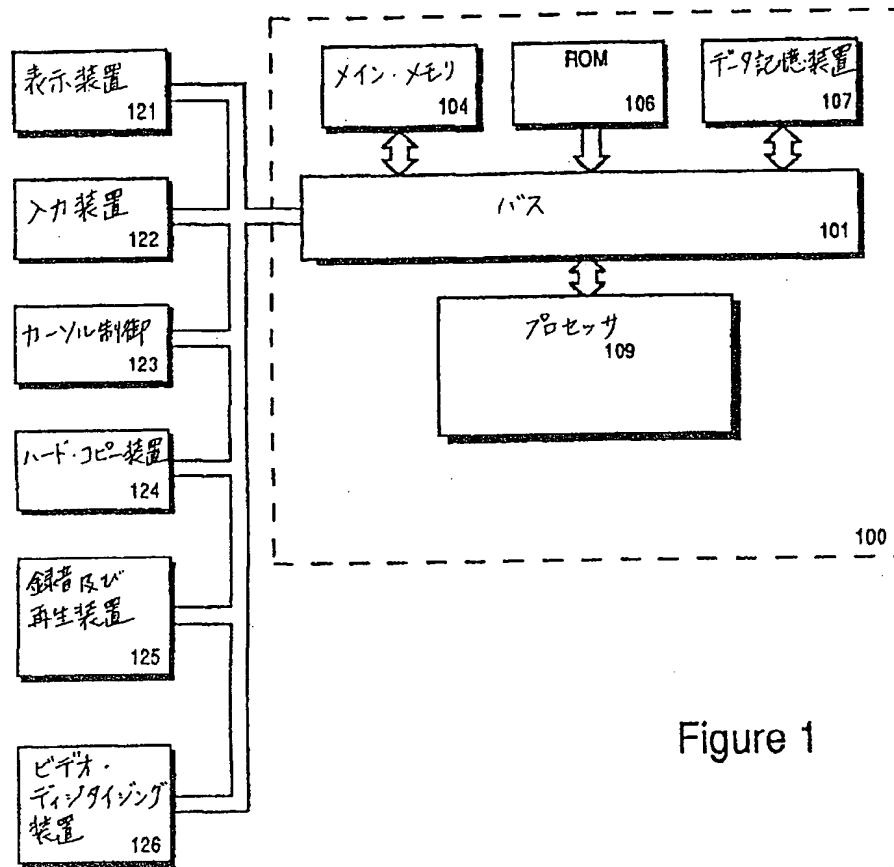


Figure 1

【図3】

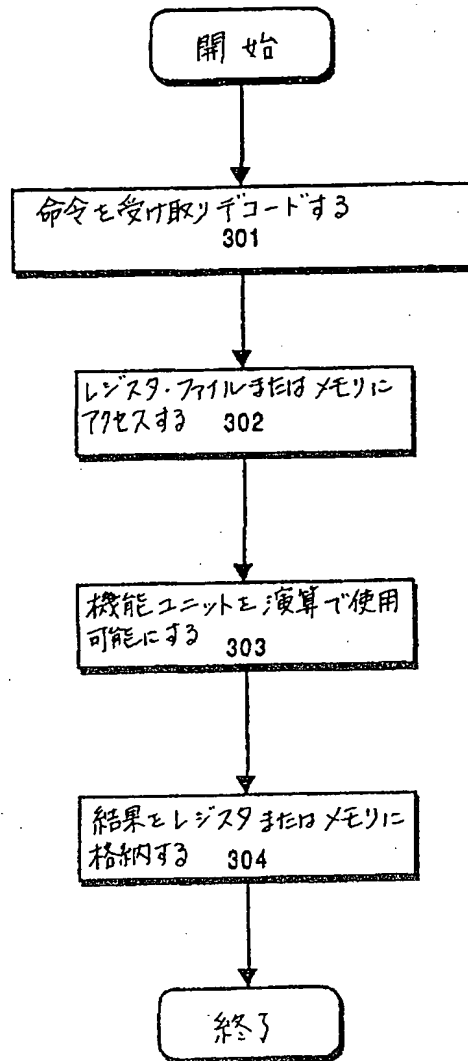


Figure 3

【図4】

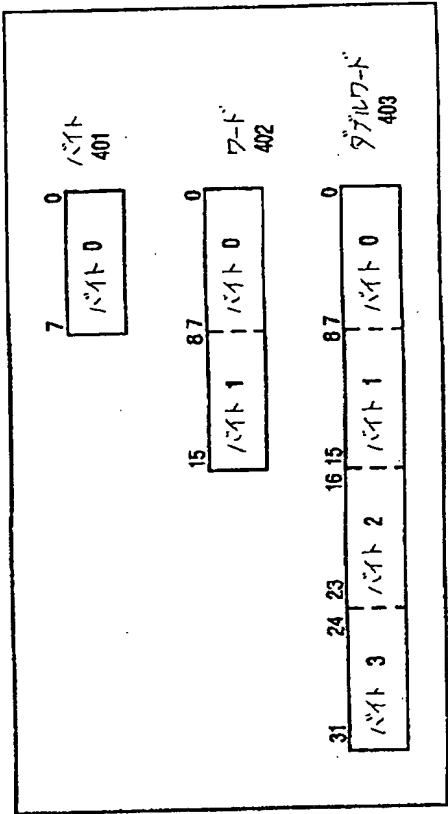


Figure 4a

【図4】

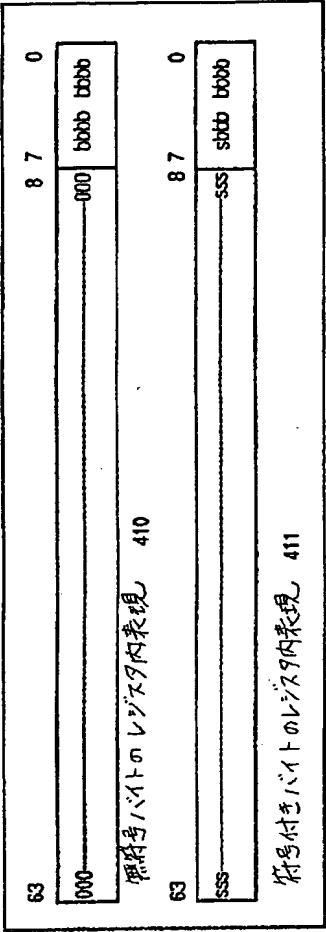


Figure 4b

【図4】

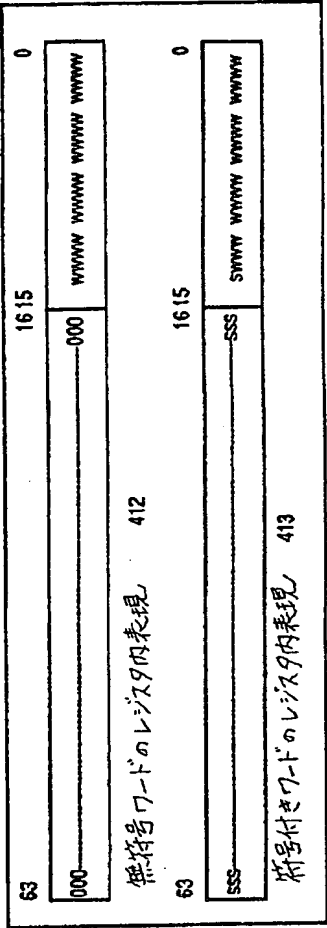


Figure 4c

【図4】

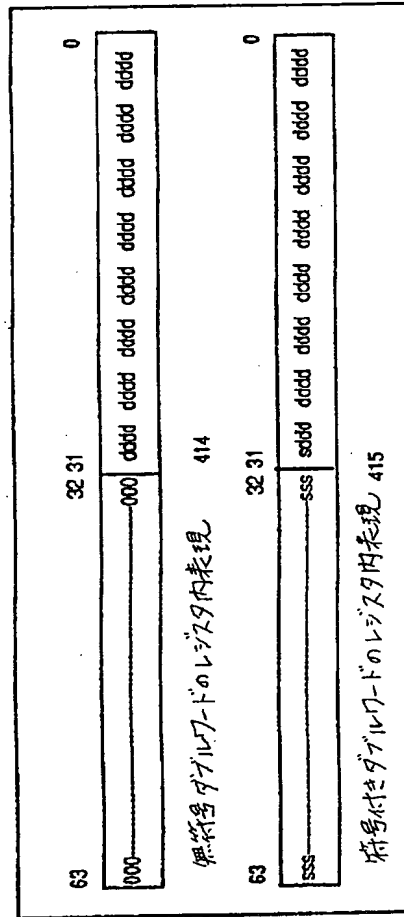


Figure 4d

【図5】

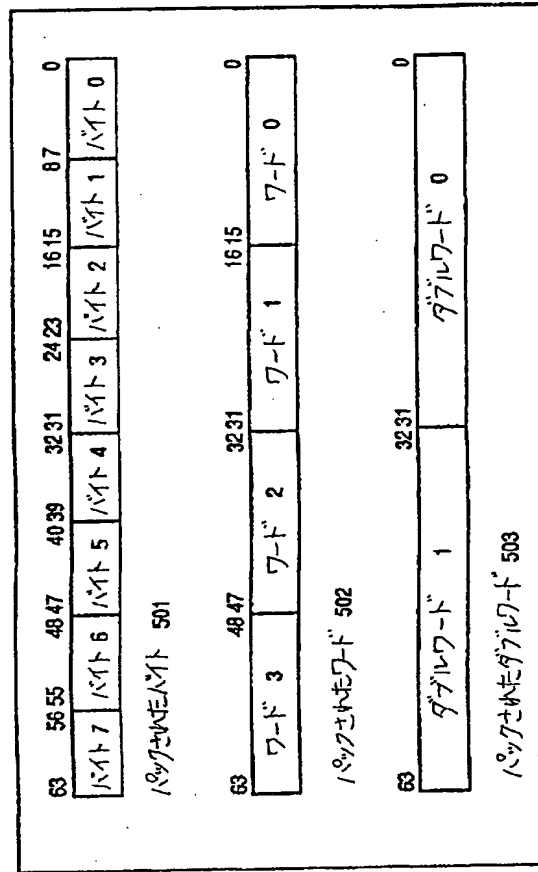


Figure 5a

63	55 54	47 46	39 38	31 30	30 22	15 14	7 6	0
bbbb	bbbb	bbbb	bbbb	bbbb	bbbb	bbbb	bbbb	bbbb
bbbb	bbbb	bbbb	bbbb	bbbb	bbbb	bbbb	bbbb	bbbb
無符号の1バイトレジスタ内表現 510								
63	55 54	47 46	39 38	31 30	30 22	15 14	7 6	0
bbbb	bbbb	bbbb	bbbb	bbbb	bbbb	bbbb	bbbb	bbbb
bbbb	bbbb	bbbb	bbbb	bbbb	bbbb	bbbb	bbbb	bbbb
符号付きの1バイトレジスタ内表現 511								

Figure 5b

【図5】

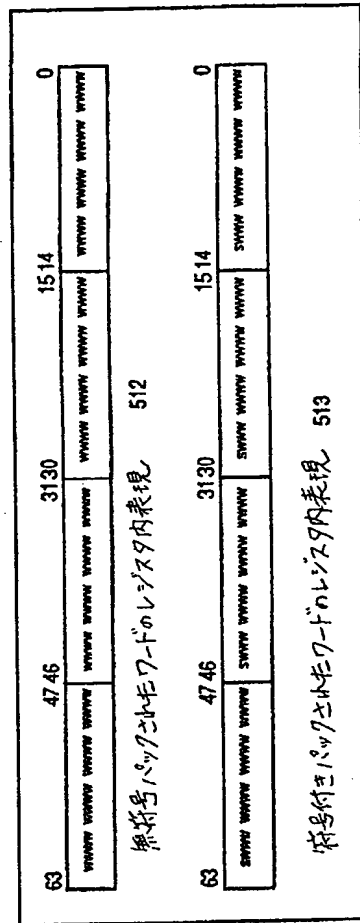


Figure 5c

【図5】

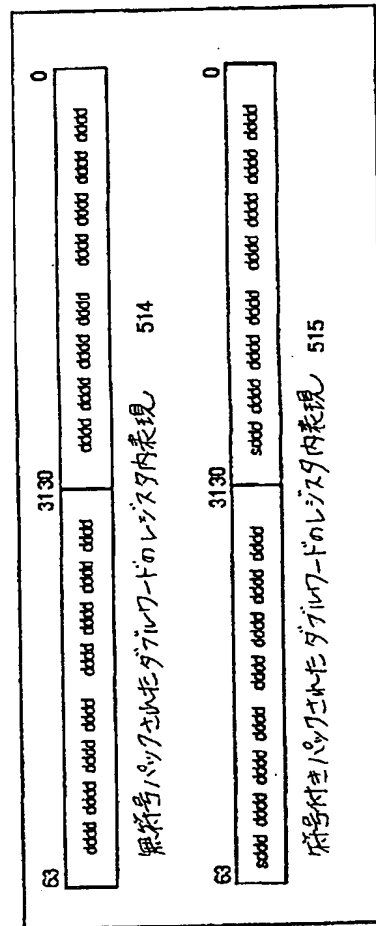


Figure 5d

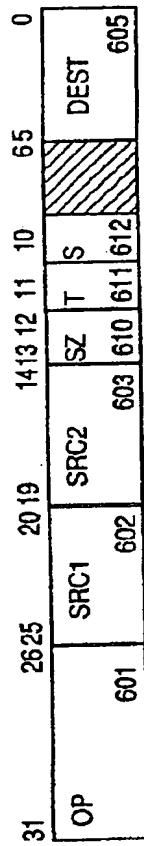


Figure 6a

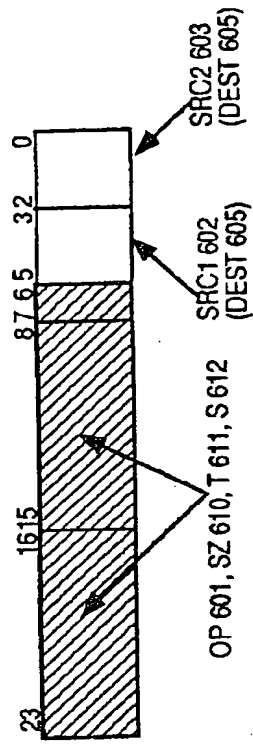


Figure 6b

【図7】

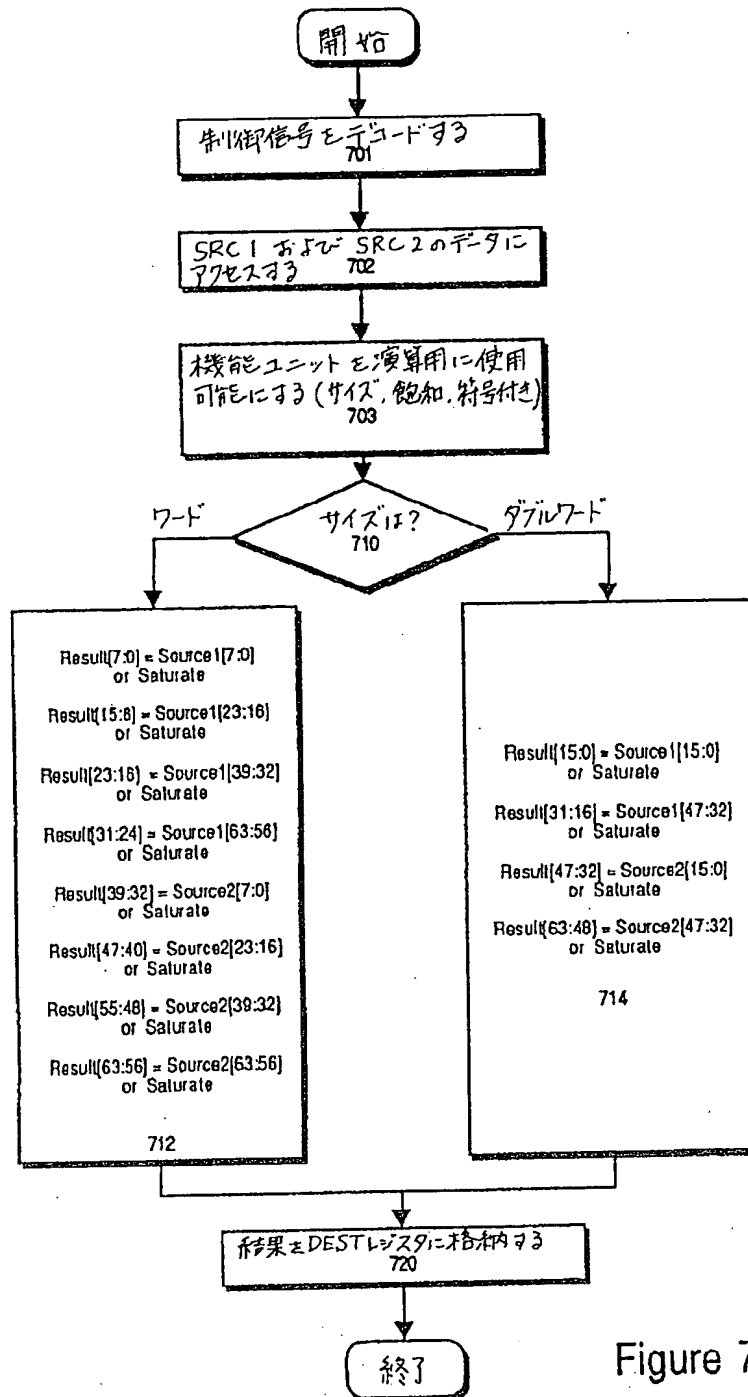


Figure 7

【図8】

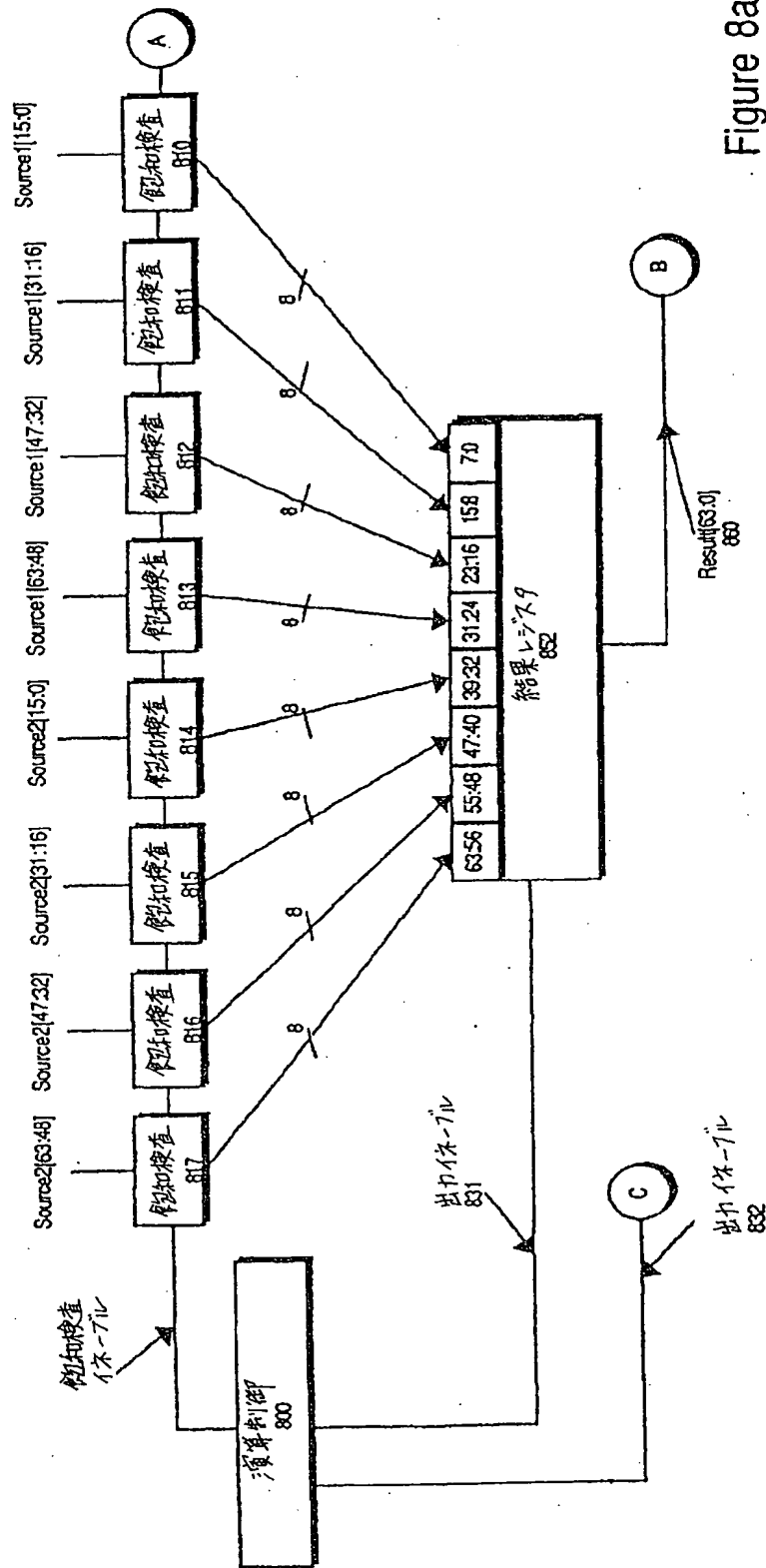


Figure 8a

【図8】

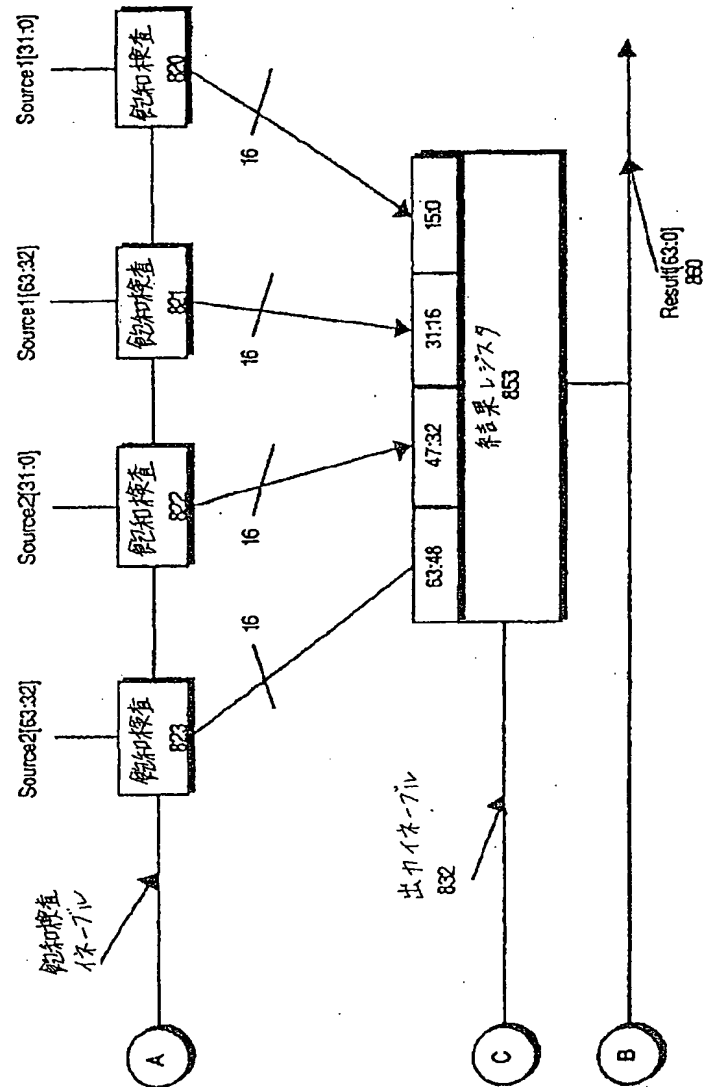


Figure 8b

【図9】

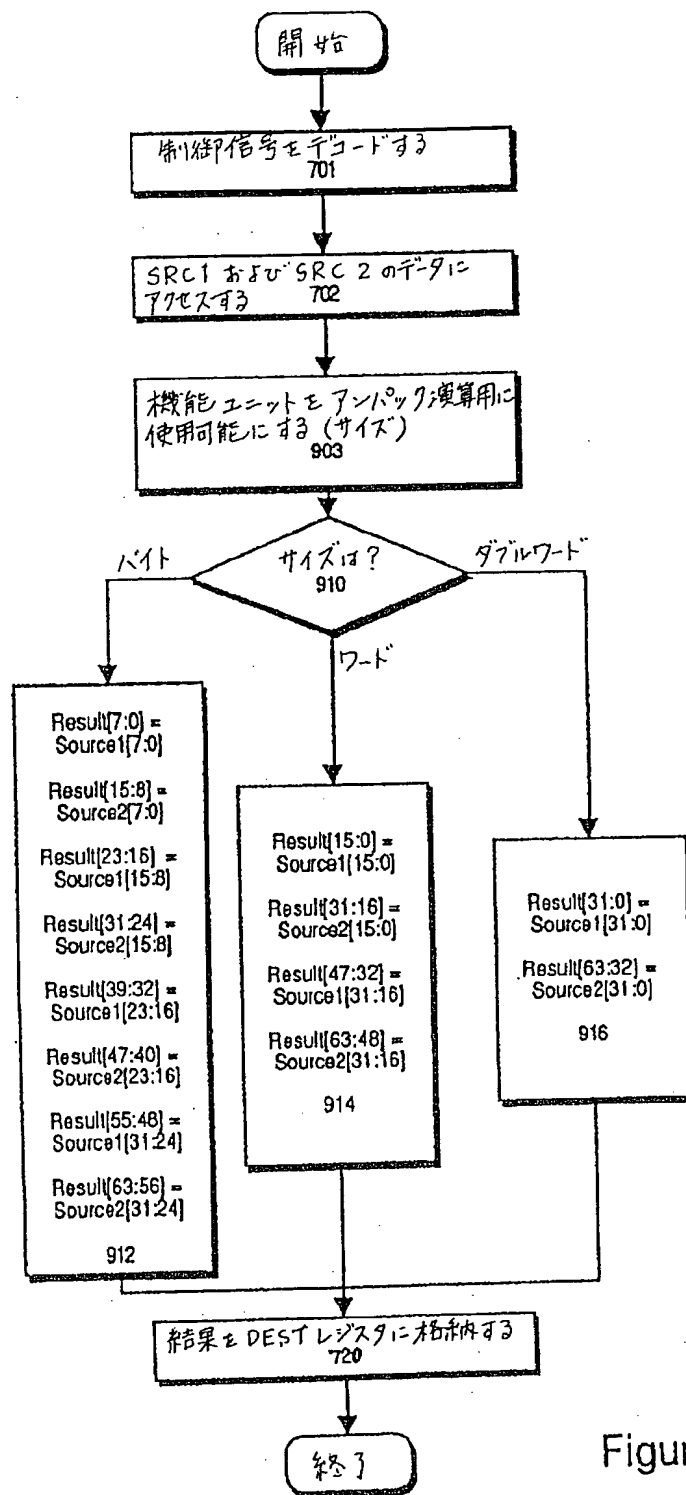


Figure 9

【図10】

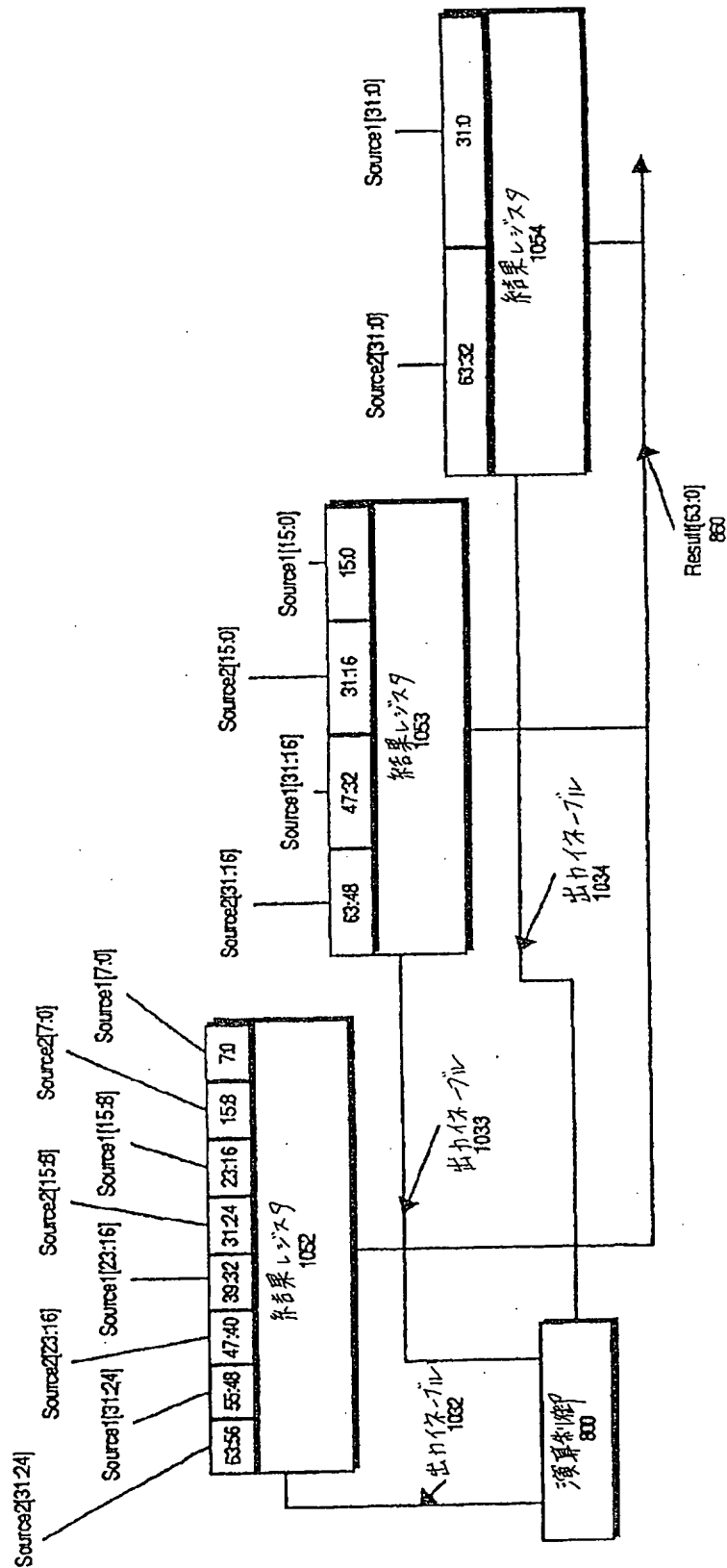


Figure 10

【手続補正書】 特許法第184条の8第1項

【提出日】 1996年9月24日

【補正内容】

補正請求の範囲

1. 第1の複数のデータ要素および第2の複数のデータ要素をそれぞれ含む第1のパックドデータおよび第2のパックドデータにして、前記第1の複数のデータ要素内の各データ要素が前記第2の複数のデータ要素内の異なるデータ要素に対応している、第1のパックドデータと第2のパックドデータを保持するよう構成された記憶域と、

アンパック命令をデコードするように構成されたデコーダと、

前記記憶域と前記デコーダとに結合され、アンパック命令に応答して、前記第1および第2の複数のデータ要素から選択された対応するデータ要素を第3のパックドデータ内の第3の複数のデータ要素として前記第1の記憶域に同時にコピーするように構成された回路と
を備えるプロセッサ。

2. 前記第1の複数のデータ要素からコピーされるデータ要素と前記第2の複数のデータ要素からコピーされるデータ要素が前記第3の複数のデータ要素として前記記憶域にインタリーブされることを特徴とする、請求項1に記載のプロセッサ。

3. 前記回路が前記第1および第2の複数のデータ要素内のデータ要素の半分を同時にコピーするように構成された、請求項2に記載のプロセッサ。

4. 前記第1の複数のデータ要素および前記第2の複数のデータ要素がそれぞれ2個、4個、または8個のデータ要素を含むことを特徴とする、請求項1に記載のプロセッサ。

5. コピーされて前記第3の複数のデータ要素を生成する前記第1の複数のデータ要素のそれぞれのデータ要素が、前記第1の複数のデータ要素が前記第1のパックドデータ内で現れる順序と同じ順序で格納されることを特徴とする、請求項2に記載のプロセッサ。

6. 第1の複数のデータ要素および第2の複数のデータ要素をそれぞれ含む第

1のパックドデータおよび第2のパックドデータにして、前記第1の複数のデータ要素内の各データ要素が前記第2の複数のデータ要素内の異なるデータ要素に対応している、第1のパックドデータと第2のパックドデータを保持するよう構成された記憶域と、

パック命令をデコードするように構成されたデコーダと、

前記記憶域と前記デコーダとに結合されパック命令に応答して、前記第1および第2の複数のデータ要素内の各データ要素の一部を第3のパックドデータ内の第3の複数のデータ要素として前記記憶域に同時にコピーするように構成された回路と

を備えるプロセッサ。

7. 前記一部が前記第1および第2の複数のデータ要素内の各データ要素内のビットの半分であることを特徴とする、請求項6に記載のプロセッサ。

8. 前記一部が前記第1および第2の複数のデータ要素内の各データ要素の下位ビットまたは上位ビットであることを特徴とする、請求項7に記載のプロセッサ。

9. 前記第1の複数のデータ要素内のデータ要素からコピーされる一部が、前記第3の複数のデータ要素内で隣接して格納されることを特徴とする、請求項8に記載のプロセッサ。

10. 前記第1および第2の複数のデータ要素からコピーされる前記一部が、前記第1および第2の複数のデータ要素が前記第1および第2のパックドデータ内で現れるのと同じ順序で格納されることを特徴とする、請求項9に記載のプロセッサ。

11. 前記第1および第2の複数のデータ要素内のすべてのデータ要素が符号付きで、前記第3の複数のデータ要素内のすべてのデータ要素が符号付きまたは無符号であることを特徴とする、請求項10に記載のプロセッサ。

12. 前記第3の複数のデータ要素のすべてのデータ要素が飽和または非飽和であることを特徴とする、請求項11に記載のプロセッサ。

13. 前記第1の複数のデータ要素と前記第2の複数のデータ要素がそれぞれ

2個、4個、または8個のデータ要素を含むことを特徴とする、請求項6に記載のプロセッサ。

【国際調査報告】

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US95/15713

A. CLASSIFICATION OF SUBJECT MATTER

IPC(6) : G06F 7/14, 9/30

US CL : 395/375, 800

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 395/375, 800

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

APS

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X,P	US, A, 5,423,010 (MIZUKAMI) 06 JUNE 1995, FIGS. 2A, 12 AND 13, COL. 3	1-14
X,P	US, A, 5,408,670 (DAVIES) 18 APRIL 1995, COL. 6, LINES 21-29	1-14
Y	US, A, 5,168,571 (HOOVER ET AL) 01 DECEMBER 1992, FIGS. 4 AND 5, COL. 6	1-14
Y	US, A, 4,139,899 (TULPUL ET AL) 13 FEBRUARY 1979, ENTIRE DOCUMENT	1-14
Y,P	US, A, 5,465,374 (DINKJIAN ET AL) 07 NOVEMBER 1995, ENTIRE DOCUMENT	1-14
A	US, A, 4,595,911 (KREGNESS ET AL) 17 JUNE 1986	1-14

☒ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

* Special categories of cited documents:	*T later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
A document defining the general state of the art which is not considered to be part of particular relevance	*X* document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
E earlier document published on or after the international filing date	*Y* documents of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
L document which may throw doubt on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	*G* document member of the same patent family
O document referring to an oral disclosure, use, exhibition or other means	
P document published prior to the international filing date but later than the priority date claimed	

Date of the actual completion of the international search

27 FEBRUARY 1996

Date of mailing of the international search report

12 MAR 1996

Name and mailing address of the ISA/US
Commissioner of Patents and Trademarks
Box PCT
Washington, D.C. 20231

Facsimile No. (703) 305-3230

Authorized officer

KENNETH S. KIM

Telephone No. (703) 305-9693

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US95/15713

C (Continuation): DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US, A, 5,276,891 (PATEL) 04 JANUARY 1994	1-14
A	US, A, 5,265,204 (KIMURA ET AL) 23 NOVEMBER 1993	1-14
A	US, A, 5,327,543 (MIURA ET AL) 05 JULY 1994	1-14
A,P	US, A, 5,390,135 (LEE ET AL) 14 FEBRUARY 1995	1-14

フロントページの続き

(81)指定国 EP(AT, BE, CH, DE, DK, ES, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OA(BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG), AP(KE, LS, MW, SD, SZ, UG), AL, AM, AT, AT, AU, BB, BG, BR, BY, CA, CH, CN, CZ, CZ, DE, DE, DK, DK, EE, EE, ES, FI, FI, GB, GE, HU, IS, JP, KE, KG, KP, KR, KZ, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SK, TJ, TM, TT, UA, UG, UZ, VN

(72)発明者 ミタル, ミリンド

アメリカ合衆国・94080・カリフォルニア
州・サウス サンフランシスコ・ヒルサイド
ブルバード・1149

(72)発明者 メネマイアー, ラリー・エム

アメリカ合衆国・95006・カリフォルニア
州・ボルダー クリーク・ピーオー ボックス・587

(72)発明者 エイタン, ベニー

イスラエル国・ハイファ・スティーブン
ウィース・25